Western Management Science Institute
University of California ● *Los Angeles*

University of California

Los Angeles

Western Management Science Institute

Working Paper No. 93

" A MARKOVIAN ALGORITHM FOR STRICTLY CONCAVE PROGRAMMING WITH LINEAR CONSTRAINTS"*

by

Arthur M. Geoffrion

March, 1966

* This work was supported partially by the Office of Naval Research under Task NR 047-041, Contract Nonr 233(75), and by the Western Management Science Institute under a grant from the Ford Foundation. The author is indebted to Joseph Naruishi for his computational assistance, which was rendered under a grant from the Division of Research of the Graduate School of Business Administration, UCLA, and to the Western Data Processing Center for making its facilities available. This paper will be presented at the Fourth International Conference on Operational Research.

ABSTRACT

Theil and van de Panne have shown how to replace the problem of maximizing a (strictly concave) quadratic function subject to linear inequality constraints by a finite sequence of sub-problems involving only linear equality constraints. In another paper, the author generalized this approach to (i) cover the case of a differentiable and strictly concave objective function, and (ii) permit almost complete flexibility in the choice of the initial sub-problem. The last feature seems essential for the approach to be of computational interest, for computational experience suggests that the number of sub-problems that must be solved and the amount of computer storage required to keep track of them have a tendency to grow approximately exponentially with the "poorness" of the choice of the initial sub-problem.

In this paper a modification of the above approach is proposed which generates the sub-problems in Markovian fashion. This all but eliminates the storage problem. Although the resulting sequence of sub-problems is no longer necessarily finite, by means of the theory of Markov chains it is shown that eventual convergence to the optimum is assured with probability one and argued that the expected number of sub-problems that must be solved increases only approximatley linearly with the "poorness" of the initial sub-problem. Computational evidence is given which supports this estimate and suggests the probable efficiency of the Markovian algorithm even for quite "bad" choices of the initial sub-problem.

This paper is a sequel to a previous one [1] in which the author gave a procedure for solving the problem

(1) Maximize $f(x)$ subject to $a_i x \leq b_i$, $i=1,\ldots,m$,

where f is a strictly concave and differentiable function that assumes its unconstrained maximum.[1/] The a_i and x are n-vectors and the b_i are scalars. It is also assumed that (1) is feasible, which implies that it has a unique optimal solution x^*, and that the a_i corresponding to the constraints that are satisfied with strict equality at x^* are linearly independent.

The procedure amounts to reducing (1) to a finite sequence of sub-problems of the form

(2) Maximize $f(x)$ subject to $a_i x = b_i$, $i \in S$,

where S is a subset of the constraint indices. Note that (2) involves only linear equality constraints, and is therefore considerably more amenable to solution than (1). The sequence of sub-problems is determined by a finite sequence $S^0, S^1, \ldots, S^K$, where S^0 is nearly arbitrary and S^K yields the optimal solution of (1). Rules are given for determining S^k given $S^0, \ldots, S^{k-1}$, and computational advantage can be taken (when (2) is solved) of the fact that S^k differs by only one constraint index from one of its predecessors.

The procedure can be viewed as a generalization of Theil and van de Panne's algorithm [2] for quadratic programming. Aside from applicability to a larger class of problems, the essential generalization is that S^0 no longer must be chosen to be the empty set. This permits advantage to be taken, by choosing S^0 appropriately, of the frequent availability of

---

1/ Linear equality constraints, which can be handled [1] by a simple modification of the procedure much more efficiently than by expressing them as inequalities, have been excluded from (1) for the sake of notational simplicity.

prior (but possibly erroneous) information regarding which of the inequality constraints of (1) are actually restrictive. In fact, with problems that have more than a few constraints it is almost mandatory to use such information to guide a propitious choice of S^o, for computational experience [1] suggests that the total number of sub-problems that must be solved and the amount of computer storage required to keep track of them tend to increase approximately exponentially with $d(S^o)$, the "distance" (to be defined more precisely below) from S^o to a "true" set of restrictive constraints of (1).

The purpose of this paper is to suggest how the approximately exponential dependence of computational work on $d(S^o)$ can be ameliorated to approximately linear dependence by generating the sub-problems in a Markovian rather than deterministic fashion. This strategy essentially eliminates the storage problem, for S^k will depend in a very simple manner only on S^{k-1} (it differs from it by exactly one constraint). It is shown that eventual termination is assured with probability 1 and argued that the expected number of sub-problems to be solved before termination should be approximately proportional to $d(S^o)$. Computational experience tends to confirm this estimate. Coefficients of proportionality of about 2 were observed, which means that for the test problems, at least, the Markovian algorithm is quite efficient even when $d(S^o)$ is large.

In what follows, the assumptions of the opening paragraph are assumed to hold. Although an effort has been made to keep the present paper self-contained at least so far as definitions are concerned, reference [1] should be consulted for motivation and proofs of the unproved assertations below.

THE MARKOVIAN ALGORITHM

Denote by B the set $\{i \in M: a_i x^* = b_i\}$ and by A the set $\{i \in M: u^*_i > 0\}$, where M is the set of the first m positive integers and the u^*_i are the usual optimal "multipliers" associated with (1). From the Kuhn-Tucker Conditions, it follows that the inclusion $A \subseteq B$ always holds. A subset S of M is said to be consistent when the linear equations $a_i x = b_i$, $i \in S$, are consistent, and independent when a_i, $i \in S$, are linearly independent.

It is known that the optimal solution x^S of (2) exists and is unique whenever S is consistent, and that $x^S = x^*$ if and only if $A \subseteq S \subseteq B$. It is convenient to denote by $d(S)$ the distance from an arbitrary subset S of M to the collection of subsets $\{S' \subseteq M: A \subseteq S' \subseteq B\}$, the metric being the number of indices in the symmetric difference set $\{A-S\} \cup \{S-B\}$. Thus $x^S = x^*$ if and only if $d(S) = 0$.

The following procedure for solving (1) is called "Markovian" because Step 2 ensures that the sequence of successive values for S constitutes a Markov chain.

Step 0: Choose any initial consistent and independent S^o, and put S equal to S^o. Go to Step 1a.

Step 1a: Solve (2) for its unique optimal solution x^S. Put u_i^S equal to 0 for $i \in M-S$ and equal to the unique solution of

$$\nabla f(x^S) = \sum_{i \in S} u_i a_i$$

for $i \in S$, where ∇ denotes the gradient operator.

If $u_i^S \geq 0$ for all $i \in S$ and $a_i x^S \leq b_i$ for all $i \in M-S$, then terminate: $(x^S, u^S) = (x^*, u^*)$. Otherwise go to Step 2a.

Step 1b: Solve the following equation for its unique solution z^S and then go to Step 2b:

$$\sum_{i \in S-i_o} z_i a_i + a_{i_o} = 0.$$

Step 2a: Choose i_o at random (with equal probability) from those i that violated the sign tests at Step 1a. If $i_o \in S$, replace S by $S-i_o$ and return to Step 1a; otherwise, replace S by $S \cup i_o$ and return to Step 1a or Step 1b according as $S \cup i_o$ is or is not consistent and independent.

Step 2b: Choose i_{oo} at random (with equal probability) from those i that satisfy $z_i^S < 0$. Replace S by $S-i_{oo}$ and return to Step 1a.

Finding (x^S, u^S) at Step 1a is equivalent to solving the Lagrange multiplier equations associated with (2). Various suggestions made in [1] for efficient computational implementation carry over here.

It follows from the results of [1] that this procedure, which differs from the original only in that a randomized rule is used to determine i_o and i_{oo}, is well-defined, and that the following lemma holds.

Lemma: At Step 2a[2/], $d(S \pm i) = d(S) - 1$ for at least one i violating a test at Step 1a. At Step 2b, $d(S-i) = d(S) - 1$ for at least one i satisfying $z_i^S < 0$.

Each time Step 1 is entered, a new iteration begins. The sequence of trial sets $\langle S^o, S^1, \ldots \rangle$ generated by the Markovian algorithm is obviously a Markov chain. The subsets of M satisfying $d(S) = 0$ can be thought of as absorbing states. In view of the random choice rule of Step 2 and the Lemma, at least one absorbing state is accessible (in exactly $d(S^o)$ transitions, in fact) from any consistent and independent S^o. By a basic property of finite Markov chains, therefore, we have the following

---

2/ $S \pm i$ denotes $S \cup i$ when $i \notin S$, and $S-i$ otherwise.

Theorem: The Markovian algorithm terminates with probability 1.[3/]

RATE OF CONVERGENCE

In applications, of course, what really matters is the distribution of the number of iterations before termination. We shall use a simple random walk model to derive an estimate of the mean of this distribution as a function of $d(S^o)$.

For any given problem (1), consider the (finite) collection of all subsets of M that could ever arise in the course of executing the Markovian algorithm. If the largest value of $d(S)$ over this collection is $D(D \leq m)$, then the collection can be partitioned naturally into $D + 1$ classes according to the value of $d(S)$ for each set. From the above discussion, it follows that the transition matrix for the associated Markov chain can be schematically represented as in Figure 1, where the natural partition has been used, the P matrices have at least one positive entry in each row, the Q matrices are unspecified, and I and 0 represent identity and null matrices. We approximate the actual situation by the simplified random walk model of Figure 2, which has $D+1$ states instead of $D+1$ classes of states. The parameter p represents the aggregate probability that a set S will transit, by an iteration of the Markovian algorithm to a set S' satisfying $d(S') = d(S)-1$.

By standard methods one can derive the mean absorption times $\bar{t}_d$ for the Markov chain represented by Fig. 2 given an intitial state $d(d=1,2,\ldots,D)$:

3/ More precisely, to every $\epsilon > 0$ there exists a positive integer N_ϵ such that the probability that termination has not occurred during the first N_ϵ iterations is less than ϵ.

	0	1	2	3	4	.	.	.	D-2	D-1	D
0	I	0	0	0	0	.	.	.	0	0	0
1	P_{10}	Q_{11}	Q_{12}	0	0	.	.	.	0	0	0
2	0	P_{21}	Q_{22}	Q_{23}	0	.	.	.	0	0	0
.						.					
.							.				
.								.			
D-1	0	0	0	0	0	.	.	.	$P_{D-1,D-2}$	$Q_{D-1,D-1}$	$Q_{D-1,D}$
D	0	0	0	0	0	.	.	.	0	$P_{D,D-1}$	$Q_{D,D}$

FIGURE 1

Transition Matrix of the Markov Chain
Associated with the Markovian Algorithm

	0	1	2	3	4	.	.	.	D-2	D-1	D
0	1	0	0	0	0	.	.	.	0	0	0
1	p	0	1-p	0	0	.	.	.	0	0	0
2	0	p	0	1-p	0	.	.	.	0	0	0
.						.					
.							.				
.								.			
D-1	0	0	0	0	0	.	.	.	p	0	1-p
D	0	0	0	0	0	.	.	.	0	p	1-p

FIGURE 2

Simplified Transition Matrix of the Markov Chain
Associated with the Markovian Algorithm

$$(3)\qquad \bar{t}_d = \begin{cases} d(2D+1) - d^2 & \text{for } p = 1/2 \\ \dfrac{d}{(2p-1)} - \left(\dfrac{1}{2p-1}\right)\left(\left(\dfrac{1-p}{p}\right)^{D-d+1} + \ldots + \left(\dfrac{1-p}{p}\right)^{D}\right) & \text{for } 0 < p \leq 1,\ p \neq \dfrac{1}{2}. \end{cases}$$

We see that $p = \frac{1}{2}$ is a key value in that, for fixed D and d, $\bar{t}_d$ increases very rapidly as p falls below 1/2 and decreases rapidly to quite small values as p rises above 1/2. For $1/2 < p \leq 1$, (3) yields a linear upper bound on $\bar{t}_d$ that is quite good for $.6 \leq p \leq 1$:

$$(4)\qquad \bar{t}_d \leq \frac{d}{2p-1} \qquad \text{for } 1/2 < p \leq 1 \text{ and } d = 1,\ldots,D.$$

Note that this upper bound does not involve D, that it has zero intercept, and that its slope is quite small for p larger than .6 or so.

This analysis suggests that, when p is greater than .5 on the average, the expected number of iterations before termination of the Markovian algorithm is approximately $d(S^o)/(2p-1)$.

COMPUTATIONAL EXPERIENCE

The Markovian algorithm was programmed for the IBM 7094 for the case in which $f(x)$ is quadratic, and tests were conducted on three medium-sized problems. Test problems 1 and 3, of practical origin, were 20 x 9 (twenty variables and 9 constraints) and 50 x 25, respectively. Test problem 2, 10 x 15, was methodically generated from a random number table. Each problem was run at 4 arbitrarily selected initial sets for each of a number of equally spaced values for $d(S^o)$, and the calculations were done in such a way as to enable p to be estimated. The estimates are .85, .84, and .78 respectively. Evidently the critical value $p = 1/2$ was amply exceeded in all of the test problems. Tables 1, 2, and 3 summarize the computational results, which tend to confirm the predicted

$d(S^0)$	Total Number of Iterations before Termination					$\frac{d(S^0)}{2\hat{p}-1}$
	Run 1	Run 2	Run 3	Run 4	Avg.	
2	2	4	2	2	2.5	2.9
4	12	6	10	6	8.5	5.7
6	6	10	8	6	7.5	8.6
8	12	10	8	8	9.5	14.3

TABLE 1

Summary of Computational Results for Test Problem 1 (20 x 9, $\hat{p}$ = .85)

$d(S^0)$	Total Number of Iterations before Termination					$\frac{d(S^0)}{2\hat{p}-1}$
	Run 1	Run 2	Run 3	Run 4	Avg.	
2	2	6	2	8	4.5	2.9
5	11	5	7	11	8.5	7.4
8	10	21	20	10	15.25	11.8
11	17	15	23	11	16.5	16.2
14	24	24	26	22	24.0	20.6

TABLE 2

Summary of Computational Results for Test Problem 2 (10 x 15, $\hat{p}$ = .84)

$d(S^0)$	Total Number of Iterations before Termination					$\frac{d(S^0)}{2\hat{p}-1}$
	Run 1	Run 2	Run 3	Run 4	Avg.	
3	5	11	15	3	8.5	5.4
8	18	8	12	30	17.0	14.3
13	27	21	27	15	22.5	23.2
18	32	32	18	22	26.0	32.2
23	27	23	35	33	29.5	41.1

TABLE 3

Summary of Computational Results for Test Problem 3 (50 x 25, $\hat{p}$ = .78)

proportional behavior for the number of iterations as a function of $d(S^o)$.

For each problem, the average computing time per iteration was well under one second.

Although computational experience with three quadratic test problems is hardly conclusive, it is remarkable that the average number of iterations should have been observed so near to the absolute minimum, which is $d(S^o)$. Perhaps variants of the simple random choice rule of Step 2 can be devised to come even closer to achieving that lower bound, as for example by weighting the probabilities in favor of constraints that are in greatest violation of a sign test.

REFERENCES

1. Geoffrion, A., "An Approach to Strictly Concave Programming with Linear Constraints," Working Paper No. 86, Western Management Science Institute, December, 1966. Submitted to SIAM Journal.

2. Theil, H. and C. van de Panne, "Quadratic Programming as an Extension of Classical Quadratic Maximization," Management Science, Vol. 7, No. 1 (October, 1960), 1 - 20.

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY *(Corporate author)*
Western Management Science Institute
University of California
Los Angeles, California 90025

2a REPORT SECURITY CLASSIFICATION
Unclassified

2b GROUP

3 REPORT TITLE
A Markovian Algorithm for strictly concave Programming with Linear Constraints

4 DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
Working Paper

5 AUTHOR(S) *(Last name, first name, initial)*
Geoffrion, Arthur M.

6 REPORT DATE	7a TOTAL NO. OF PAGES	7b NO OF REFS
March 1966	10	2

8a CONTRACT OR GRANT NO
Nonr 233(75)
b PROJECT NO
047-041
c
d

9a ORIGINATOR'S REPORT NUMBER(S)
Working Paper No. 98

9b OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10 AVAILABILITY/LIMITATION NOTICES
Available upon request: Western Management Science Institute
University of California
Los Angeles, California 90025

11 SUPPLEMENTARY NOTES

12 SPONSORING MILITARY ACTIVITY

13 ABSTRACT

Theil and van de Panne have shown how to replace the problem of maximizing a (strictly concave) quadratic function subject to linear inequality constraints by a finite sequence of sub-problems involving only linear equality constraints. In another paper, the author generalized this approach to (i) cover the case of a differentiable and strictly concave objective function, and (ii) permit almost complete flexibility in the choice of the initial sub-problem. The last feature seems essential for the approach to be of computational interest, for computational experience suggests that the number of sub-problems that must be solved and the amount of computer storage required to keep track of them have a tendency to grow approximately exponentially with the "poorness" of the choice of the initial sub-problem.

In this paper a modification of the above approach is proposed which generates the sub-problems in Markovian fashion. This all but eliminates the storage problem. Although the resulting sequence of sub-problems is no longer necessarily finite, by means of the theory of Markov chains it is shown that eventual convergence to the optimum is assured with probability one and argued that the expected number of sub-problems that must be solved increases only approximately linearly with the "poorness" of the initial sub-problem. Computational evidence is given which supports this estimate and suggests the probable efficiency of the Markovian algorithm even for quite "bad" choices of the initial sub-problem.

DD FORM 1 JAN 64 1473 *0101-807-6800*

Security Classification

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Concave programming **Markov chains** **Quadratic programming**						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2*a*. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2*b*. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7*a*. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7*b*. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8*a*. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8*b*, 8*c*, & 8*d*. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9*a*. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9*b*. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through

_____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through

_____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through

_____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13 ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, or *(U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, roles, and weights is optional.